TAGE-SC-L Branch Predictors Again

André Seznec

TAGE-SC-L Branch Predictors Again*

André Seznec
INRIA/IRISA

Outline

In this study, we explore the performance limits of these TAGE-SC-L predictors for respectively 8Kbytes and 64Kbytes of storage budget.

For a 8KB storage budget, our submitted predictor used most of its storage budget on the TAGE predictor, features a very small loop predictor LP and a neural statistical corrector exploiting global history path and very limited local history. The submitted 8Kbytes predictor achieves **4.991** MPKI on the CBP-5 train traces.

With a larger storage budget, one can invest more significant storage budget in the adjunct predictors. The submitted 512Kbits TAGE-SC-L predictor features a TAGE predictor, a loop predictor LP and a quite complex (57 Kbits) neural statistical corrector that exploits various form of local histories, global branch history, IMLI counter [12] . The submitted 64KB TAGE-SC-L predictor achieves **3.986** MPKI on the CBP-5 train traces.

1 General view of the TAGE-SC-L predictor

The TAGE-SC-L predictor consists of three components: a TAGE predictor, a statistical corrector predictor and a loop predictor (Figure 1). The TAGE predictor provides the main prediction. Then this prediction is used by the statistical corrector predictor which role consists in confirming (general case) or reverting the prediction. The statistical predictor reverts the prediction when it appears that, in similar circumstances (prediction, branch histories, branch confidence, ..) TAGE statistically mispredicted. The loop predictor is (marginally) useful to predict regular loops with long loop bodies.

2 The TAGE conditional branch predictor

The TAGE predictor was described in [11] and [6]. Only marginal modifications are introduced in this study. Figure 2 illustrates a TAGE predictor. The TAGE predictor features a base predictor T0 in charge of providing a basic prediction and a set of (partially) tagged predictor components Ti. These tagged predictor components Ti, $1 \leq i \leq M$ are indexed using different history lengths that form a geometric series [4], i.e, $L(i) = (int)(\alpha^{i-1} * L(1) + 0.5)$.

*This work was partially supported by an Intel research grant

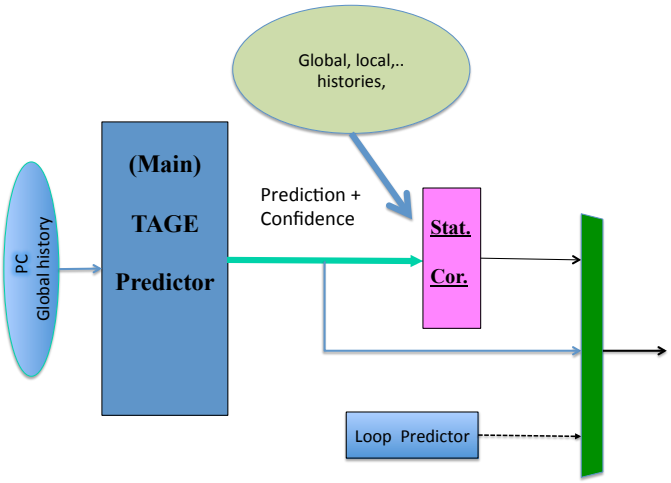

Figure 1. The TAGE-SC-L predictor: a TAGE predictor backed with a Statistical Corrector predictor and a loop predictor

Throughout this paper, the base predictor will be a simple PC-indexed 2-bit counter bimodal table; in order to save storage space, the hysteresis bit is shared among several counters as in [5].

An entry in a tagged component consists in a signed counter *ctr* which sign provides the prediction, a (partial) tag and an unsigned useful counter *u*. In our study, *u* is a single bit for the 64 KB predictor and a 2-bit counter for the 8KB predictor. The prediction counter *ctr* is a 3-bit counter.

Prediction computation and predictor updates are extensively described in previous work.

The global history vector The global branch history vector consists 2-bit for direct branches and 3-bit for indirect branches. It combines path and direction.

2.1 Special tricks for the championship

On the allocation policy

- For the 64KB predictor, *u* is only 1-bit. The global decrement of all the *u* leaves every entry unprotected. We use a simple trick to allow entry with medium and high confidence to remain partially protected. Only entries with $u = 0$ and $|2 * ctr + 1| < 5|$ can be replaced,

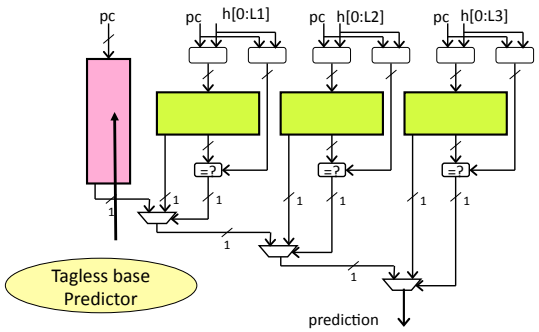

Figure 2. The TAGE predictor

otherwise if $u = 0$ then we decrease $|2 * ctr + 1|$. This trick nearly fills the gap between using 1-bit and 2-bit u.

- For the 8KB predictor, in many cases a recently allocated entry has a too short presence in the predictor to be reused before replacement. To decrease this impact, we use a DIP-like allocation [3], i.e. from time to time, u is set as 1 instead 0. *This trick has marginal impact: 0.2 % reduction of the misprediction rate.*

Bank interleaving In order to share the storage space within the predictor, we use bank-interleaving. The small history lengths share one group of banks. The long history lengths share another group of banks. This allows to use different tag widths for short and long history lengths.

Partial associativity Using associativity for the predictor tables helps marginally, but on the other side increases the false tag match rate. We found that the good tradeoff is to use 2-associativity only for the intermediate history lengths. This reduces the misprediction rate by 0.6%.

3 The loop predictor component

The loop predictor simply tries to identify regular loops with constant number of iterations.

The implemented loop predictor is copied from [10]. The loop predictor features only a limited number of entries, 8 for the 8KB predictor and 32 for the 64KB predictor.

The loop predictor reduces the misprediction rate by approximately 0.3% on both predictors.

4 Statistical Corrector Predictor

TAGE fails at predicting statistically biased branches e.g. branches that have only a small bias towards a direction, but are not strongly correlated with the global history path.

In [7, 8], we introduced the Statistical Corrector predictor to better predict this class of statistically biased branches, The correction aims at detecting the unlikely predictions and to revert them. The prediction flowing from TAGE as well as information on the branch (address, global history, global path, local history) are presented to Statistical Corrector predictor which decides whether or not to invert the prediction. Since in most cases the prediction provided by the TAGE predictor is correct, the Statistical Corrector predictor can be quite small.

For the submitted predictors, we use the Multi-GEHL statistical corrector used in [10]. This predictor component is a neural-based predictor component and can accomodate all variations of branch and path history: global and local.

4.1 General parameters

All tables are 6-bit counters. The prediction is computed as the sign of the sum of the (centered) predictions read on all the Statistical Corrector tables.

The MGSC predictor tables are updated using a dynamic threshold policy as suggested for the GEHL predictor [4]. *As suggested in [2], a PC-indexed table of dynamic threshold is used (32 entries), enabling marginal benefit (0.1% misprediction decrease).*

4.2 The Multi-GEHL Statistical Corrector Predictor for the 8KB predictor

For the 8 KB predictor, the MGSC predictor includes:

- A Bias component: 3 tables indexed through the PC, the TAGE predicted direction + the confidence in the TAGE prediction and the number of the highest matching table.
- 3 GEHL-like components respectively indexed using the global conditional branch history (2 tables), the global history of the backward branches (2 tables) and a 64-entry local history (2 tables). The GEHL components do not feature a PC indexed table.

4.3 The Multi-GEHL Statistical Corrector Predictor for the 64KB predictor

For the 64 KB predictor, the MGSC predictor includes:

- A Bias component (similar as above)
- 7 GEHL-like components respectively indexed using the global conditional branch history (2 tables), the global history of the backward branches (2 tables), an IMLI counter indexed table, another IMLI-based GEHL predictor ("constant IMLI" branch history) and 3 local history GEHL components respectively with 256-entry, 16-entry and 16-entry local history. The GEHL components do not feature a PC indexed table.

The use of two distinct local histories was introduced in [1]. The benefit of the 2nd local history is marginal (about 0.5 % misprediction reduction). However adding a 3rd local history table with also 16 history entries, with a different hashing function on the local history table induces an extra accuracy benefit in the same range.

4.4 Small tricks

As suggested in [2], a PC-indexed table of dynamic threshold is used (resp. 32 and 64 entries), enabling marginal benefit (0.2% misprediction decrease).

In the prediction computation, each of the GEHL components is affected a dynamic multiplicative factor (1 or 2). This factor is determined at runtime through monitoring the usefulness of the GEHL component (resp. 4 and 8 counters).

5 Selecting between TAGE and statistical corrector outputs

The prediction flowing out from the statistical corrector is generally more accurate than the TAGE prediction. However using a very simple chooser results in a slightly better accuracy than just using the statistical corrector output.

When the output of MGSC is **very low confidence** (i.e. $|sum| < threshold/2$), TAGE might be a little bit more accurate than MGSC. In the general case we select the MGSC output, apart 1) when TAGE output is **high confidence** (see [9]) and MGSC output is very low confidence then the prediction is chosen based on MGSC sum value and a monitoring counter, 2) when TAGE output is **medium confidence** (see [9]) and MGSC output is *very very low confidence*, (i.e. $|sum| < threshold/4$), the predictionis chosen based on a second monitoring counter.

This simple policy enables about 0.7% misprediction reduction.

6 Predictor storage budget

6.1 The 8KB TAGE-SC-L predictor

The TAGE component consists in:

- a (4Kbits prediction, 1Kbits hysteresis) base predictor
- two bank-interleaved tagged tables featuring respectively 9 13-bit 128-entry banks, and 17 17-bit 128-entry banks. The respective tag width are 8 and 12 bits.
- a 1000 bits global history length, a 27 bits global path length.
- 8 5-bit USEALT counters,
- a 10-bit counter for monitoring the allocation policy.

The total storage budget of TAGE is 58,165 bits.

The loop predictor budget is only 312 bits (8 x 39 bits).

The statistical corrector budget is 8,872 bits:

- a total of 10 6-bit 128-entry tables
- 64 8-bit threshold counters + a global 12 bit threshold counter.
- 5 x 8 6-bit counters for the *dynamic multiplicative factors*
- 64 6-bit local history, 16-bit global branch history, 8-bit IMLI counter, 16-bit global backward branch history,
- 2 counters for the chooser.

The total budget of the submitted predictor is: 67,349 storage bits.

6.2 The 64KB TAGE-SC-L predictor

The TAGE component consists in:

- a (8Kbits prediction, 2Kbits hysteresis) base predictor
- two bank-interleaved tagged tables featuring respectively 10 12-bit 1K-entry banks, and 20 16-bit 1K-entry banks. The respective tag widths are 8 and 11 bits.
- a 3000 bits global history length, a 27 bits global path length.
- 16 5-bit USEALT counters,
- a 10-bit counter for monitoring the allocation policy.

The total storage budget of TAGE is 463,917 bits.

The loop predictor budget is only 1248 (32 x 39 bits).

The statistical corrector budget is 58,190 bits:

- a total of 2 1K-entry tables, 8 512-entry tables, 9 256-entry tables and 1 128-entry table. All entries are 6-bit counters
- 64 8-bit threshold counters + 1 12-bit global threshold counter.
- 8 x 8 6-bit counters for the *dynamic multiplicative factors*.
- 256 11-bit local histories, 16 16-bit local histories, 16 9-bit local history, a 40-bit global branch history, a 16-bit global backward branch history, a 8-bit IMLI counter, and 256 "constant IMLI" 10-bit histories.
- 2 counters for the chooser.

The total budget of the submitted predictor is 523,355 storage bits.

7 Performance analysis of the submitted predictors

The submitted predictors are directly derived from the predictors that won the CBP4 championship. Porting the CBP4 256Kbits and 32Kbits winner and just doubling their storage size lead to a performance of 4.155 MPKI and 5.402 MPKI, respectively. We managed to outperform these ports by respectively more than 3% and more than 6%.

The benefit comes from several factors. The most significant benefit was obtained through bank-interleaving the TAGE predictor and using partial associativity for the medium history lengths. This respectively leads to 2% and 3% misprediction reduction on the large and small predictor respectively.

For the small predictor, doubling the storage budget allows to implement a moderately complex SC predictor that captures quite significant correlation than the predictor in CBP4.

For the large predictor, a lot a minor improvements on the SC component, allowed to improved the overall benefit of the SC component from 6.5 % on the CBP4 predictor to 8 % on our submissions. E.g. using the extra information flowing from the TAGE predictor (confidence, number of the predicting table) enhances the quality of the prediction made by the bias tables. This enhances the prediction by 0.3 %; IMLI-based table improves by 0.2 %; global backward branch history brings 0.3 % when replacing global path history etc.

8 Conclusion

The TAGE-SC-L predictor can be adapted to various predictor storage budgets.

For small storage budgets (e.g. 8KB), large accuracy benefit is obtained through increasing the number or the sizes of the TAGE tagged tables, therefore only very small loop predictors and statistical correctors can be used. However, with a minimal budget, the statistical corrector brings about 6 % accuracy benefit.

For larger storage budgets, the adjunct predictors can implement more complex schemes as illustrated by the multi-GEHL statistical corrector used for the 64 KB predictor, bringing about 8% misprediction reduction.

The submitted predictors are optimized to achieve ultimate accuracy at the fixed storage budgets for the championship. This leads to use unrealistic number of tables, tag widths and many different schemes in the multi-GEHL statistical corrector. However, simpler designs with storage budgets in the same range, but using much smaller number of tables would achieve prediction accuracy in the same range.

References

[1] Y. Ishii, K. Kuroyanagi, T. Sawada, M. Inaba, and K. Hiraki. Revisiting local history for improving fused two-level branch predictor. In *Proceedings of the 3rd Championship on Branch Prediction, http://www.jilp.org/jwac-2/*, 2011.

[2] Daniel A. Jiménez. Oh-snap: Optimized hybrid scaled neural analog predictor. In *Proceedings of the 3rd Championship on Branch Prediction, http://www.jilp.org/jwac-2/*, 2011.

[3] Moinuddin K. Qureshi, Aamer Jaleel, Yale N. Patt, Simon C. Steely Jr., and Joel S. Emer. Adaptive insertion policies for high performance caching. In *34th International Symposium on Computer Architecture (ISCA 2007), June 9-13, 2007, San Diego, California, USA*, pages 381–391, 2007.

[4] A. Seznec. Analysis of the O-GEHL branch predictor. In *Proceedings of the 32nd Annual International Symposium on Computer Architecture*, june 2005.

[5] A. Seznec, S. Felix, V. Krishnan, and Y. Sazeidès. Design tradeoffs for the ev8 branch predictor. In *Proceedings of the 29th Annual International Symposium on Computer Architecture*, 2002.

[6] André Seznec. The L-TAGE branch predictor. *Journal of Instruction Level Parallelism (http://wwwjilp.org/vol9)*, May 2007.

[7] André Seznec. A 64 kbytes ISL-TAGE branch predictor. In *Proceedings of the 3rd Championship Branch Prediction*, June 2011.

[8] André Seznec. A new case for the tage branch predictor. In *Proceedings of the MICR0 44*, 2011.

[9] André Seznec. Storage Free Confidence Estimation for the TAGE branch predictor. In *Proceedings of the 17th IEEE Symposium on High-Performance Computer Architecture (HPCA 2011)*, Feb 2011.

[10] André Seznec. Tage-sc-l branch predictors. In *Proceedings of the 4th Championship on Branch Prediction, http://www.jilp.org/cbp2014/*, 2014.

[11] André Seznec and Pierre Michaud. A case for (partially)-tagged geometric history length predictors. *Journal of Instruction Level Parallelism (http://www.jilp.org/vol8)*, April 2006.

[12] André Seznec, Joshua San Miguel, and Jorge Albericio. The inner most loop iteration counter: a new dimension in branch history. In *Proceedings of the 48th International Symposium on Microarchitecture, MICRO 2015, Waikiki, HI, USA, December 5-9, 2015*, pages 347–357, 2015.